Índice espacio-temporal para redes móviles. Diseño de estructuras de datos y algoritmos de consultas.

Daniela C. Giraudi, Gabriela S. Segura Guzmán, Edilma O. Gagliardi
Universidad Nacional de San Luis, Facultad de Ciencias Físico,
Matemáticas y Naturales, Departamento de Informática
San Luis, Argentina, D5700HHW
{dcgiraud, ggsegura, oli}@unsl.edu.ar

Gregorio Hernández Peñalver
Universidad Politécnica de Madrid, Facultad de Informática
Madrid, España
gregorio@dma.fi.upm.es

Resumen

Nuestro trabajo de investigación consiste en proveer la manera en que dada una red, total o parcialmente conocida, modelada como un grafo geométrico, sobre la que se ubican objetos en movimiento, sea posible encontrar una ruta desde una posición origen a una posición destino mediante diversas heurísticas, basadas en conocimientos parciales de la red subyacente, con la característica principal de considerar la definición de la red según el índice espacio-temporal I+MON-Tree [CO06].

Para ello, proponemos una primera aproximación de la estructura de datos necesaria capaz de almacenar información de los objetos que se mueven sobre la red y la nueva disposición si la red cambia.

Palabras claves: Estructuras de datos, redes móviles, Geometría Computacional.

1. INTRODUCCIÓN

El índice espacio-temporal I+MON-Tree permite almacenar y recuperar información histórica y actual de objetos en movimiento sobre redes fijas. Este método de acceso indexa objetos que se mueven sobre redes conocidas, asumiendo que los mismos tienen la capacidad de informar las coordenadas y el tiempo en el que arriban a una nueva posición. Esta estructura, además de mantener información histórica de los objetos, tiene la capacidad de almacenar y consultar información de la posición actual de los mismos. Este método surgió por la necesidad de que ciertas aplicaciones requieren conocimiento acerca de la última posición en la que el objeto se encuentra y/o por la necesidad de conocer la trayectoria del mismo.

Proponemos un posible diseño de extensión a la estructura I+MON-Tree en relación a las redes móviles. De este modo, se pretende mantener registro de los objetos en movimiento que exploran su ambiente de trabajo a medida que van descubriendo la ruta hacia su destino, basándose en diversas heurísticas de búsqueda que están delineadas en función de las poligonales (rutas) definidas sobre la red.

En el inicio de nuestra investigación, estudiando el índice espacio-temporal I+MON-Tree, observamos que los autores de MON-Tree [AG04a] para la evaluación experimental y generación de lotes de prueba utilizaron el generador de objetos en movimiento [BRINK00], basado en la red de carreteras de Alemania. Debido a las dificultades ocurridas en el trabajo de experimentación de I+MON-Tree, surgió la necesidad de crear nuestros propios lotes de prueba y de cambiar el algoritmo de ruteo usados por MON-Tree e I+MON-Tree, encaminando nuestra investigación

central hacia este tema y su evaluación sobre redes móviles.

Considerando el lineamiento de nuestro trabajo, surgió la necesidad de proponer una posible adaptación del índice I+MON-Tree a redes móviles. Por ello, presentamos una primera versión de diseño que permitiría considerar dichas redes. A esta propuesta la hemos llamado *I+MON-Tree-RM*.

Como ya planteamos anteriormente, nos abocaremos a una propuesta de diseño, y dejamos para futuros trabajos de investigación y desarrollo su correspondiente refinamiento, implementación y evaluación experimental del buen desempeño de la estructura.

2. I+MON-Tree-RM

De acuerdo a la definición del I+MON-Tree, existen dos tipos diferentes de modelos de redes que indexa este índice. El primer modelo es *orientado a arcos*, es decir, la red se compone de aristas y vértices, donde cada arista tiene asociada una poligonal. El segundo modelo está *orientado a rutas*, es decir, la red se compone de un conjunto de rutas y un conjunto de intersecciones entre esas rutas. De todo esto, rescatamos fundamentalmente la posibilidad de determinar poligonales constituidas sobre la red. Entonces, en lo subsiguiente nos abocaremos a considerar dichas poligonales, y no haremos discriminación por modelo.

Para comprender un poco más, veamos el siguiente ejemplo. Supongamos que el objeto *o* transita sobre la red con destino al vértice *d*. Es claro que, en el camino hacia su destino éste puede atravesar una o varias poligonales. Cuando el objeto arriba a un vértice, éste informa su llegada de manera que se registra el instante de tiempo en que arribó a ese punto. De acuerdo a este registro y conociendo cuál es su próximo destino inmediato, podemos conocer cuál es la posición relativa del mismo. Veamos la siguiente figura:

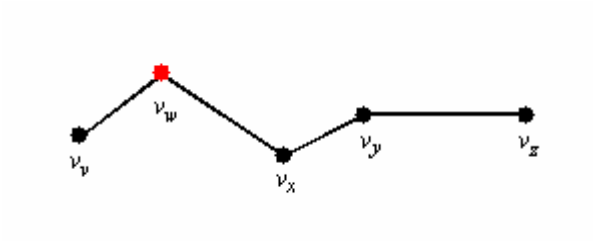

Figura 1: El objeto *o* arriba al vértice v_w en un instante de tiempo

En la figura vemos que el objeto se registra en el vértice v_w y transita sobre esta poligonal en dirección al vértice inmediato v_x.

Los R-Tree´s inferiores indexan el movimiento de los objetos que transitan sobre las poligonales que constituyen la red. Este movimiento se representa por medio de un intervalo de posición (p_1; p_2) y un intervalo de tiempo (t_1; t_2), donde 0 representa el instante de tiempo en que el objeto estuvo en la posición p_1 y 1 representa el instante de tiempo en que el objeto estuvo en la posición p_2. Así, p_1 y p_2 almacenan la posición relativa del objeto dentro de la poligonal en el instante t_1 y t_2, respectivamente.

Es decir, el objeto *o* registró su arribo en el vértice v_w en el instante t_i. En un instante de tiempo t_j, donde $i < j$, siendo que el objeto se dirige hacia el vértice v_x podemos conocer su posición relativa.

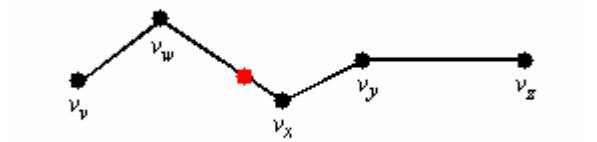

Figura 2: El objeto transita sobre la poligonal

Como vemos en la figura anterior, el objeto transita sobre la poligonal. En ese instante de tiempo no se conoce cuál es la posición física del objeto, sino que podemos conocer cuál es la posición relativa respecto del último arribo. Esto significa, que conocemos el porcentaje de movimiento que efectuó el objeto sobre una parte de la poligonal, a partir del último vértice al que arribó. Esta información se almacena en el R-Tree correspondiente a dicha poligonal, donde se almacena el registro de los movimientos de cada uno de los objetos que transitan sobre ella.

Por otro lado, el índice I+MON-Tree está formado por un Índice de Información Histórica, para almacenar los estados pasados de los objetos; y, de un Índice de Información Actual, para almacenar los estados actuales de los objetos cuyo instante final de permanencia en una posición aún no ha sido definido.

También, debemos tener en cuenta que la red está expuesta a posibles cambios, por lo que esta nueva información debe ser registrada en el índice. En este sentido, nosotros consideramos los cambios desde un punto de vista más general, dirigido a una poligonal, más que a un vértice en particular. Por lo tanto, debemos emplear alguna política de almacenamiento. En la Figura 1, mostramos un posible diseño de extensión a la estructura I+MON-Tree en relación a las redes móviles.

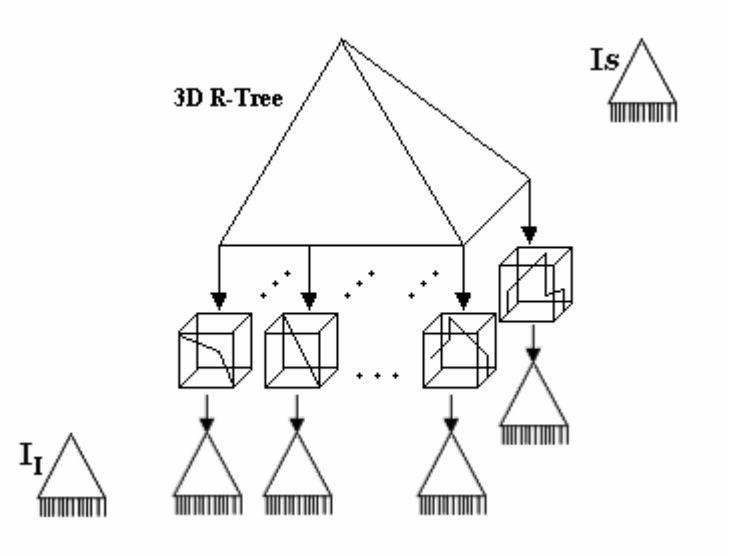

Figura 3: I+MON-Tree-RM

En este sentido, nosotros cambiamos el 2D R-Tree superior de la estructura por un I+3 R-Tree [Cgg06] de manera tal de poder mantener los cambios producidos en las poligonales a través del tiempo.

En cada R-Tree inferior del índice I+MON-Tree se almacena información de los objetos que atraviesan la red en un momento dado. En este índice, la red es fija, de manera que, la posición de cada uno de todos los puntos que constituyen las poligonales de la red subyacente se mantienen de manera constante a través del tiempo. En el caso de una red móvil, la misma está sujeta a cambios. Por lo que, el índice debe ser capaz de registrar las modificaciones efectuadas en la red. Para ello proponemos el siguiente diseño:

- El *3D R-Tree* almacena los *cubos definidos*, cada uno de los cuales representa la estadía de la poligonal en una posición durante un intervalo definido de tiempo. Es decir, toda la información espacio – temporal histórica se mantiene en el 3D R-Tree. Las nuplas almacenadas en el 3D R-Tree son del tipo <*pid, mbr_3D, p_cubos*> donde:
  - *pid*: código identificador de la poligonal;
  - *mbr_3D*: región tridimensional cuya altura representa el intervalo temporal durante el cual la

poligonal se mantuvo en la posición espacial definida por su base;

- *p_cubos*: puntero al cubo anterior correspondiente al mismo pid, utilizado para mantener un historial de trayectoria de la poligonal.

- En el índice I_S se almacenan los *cubos abiertos*. Esto es aquellos cubos para los cuales su techo aún no está definido. También se guardan las referencias necesarias a los cubos anteriores que describen los cambios de posiciones de las poligonales.

El índice I_S es una lista secuencial de N elementos, donde N es la cantidad de poligonales consideradas, que almacena nuplas de la forma <*pid, mbr, t, p3D, pa, ps*> donde:

- *pid*: código identificador de la poligonal;
- *mbr*: región aproximada que ocupa actualmente la poligonal;
- *t*: tiempo en que la poligonal cambio sus posiciones a la ubicación actual;
- *p3D*: puntero al cubo anterior correspondiente al mismo pid, utilizado para mantener los cambios físicos de las poligonales;
- *pa*: puntero a la poligonal insertada en el instante de tiempo inmediatamente anterior; y
- *ps*: puntero a la poligonal insertada en el instante de tiempo siguiente.

Los punteros *pa* y *ps* son utilizados para enlazar los tiempos en orden creciente.

La idea subyacente es mantener en la estructura I+3 R-Tree todas las modificaciones que se producen en la red, a nivel de poligonales. De esta forma, resulta posible consultar las distintas posiciones de las mismas definidas sobre la red.

En la siguiente figura observamos los cambios físicos de una poligonal en diferentes instantes de tiempo, así es que, cada cubo que representa la poligonal en el instante de tiempo t_i, tiene asociado un R-Tree inferior con todos aquellos objetos que transitaron por esa poligonal en el tiempo t_i.

Cuando consultamos en esta estructura, primeramente accedemos a la estructura I_s, recordemos que esta mantiene la ubicación actual de la poligonal con un puntero que permite acceder a las ubicaciones anteriores de la misma.

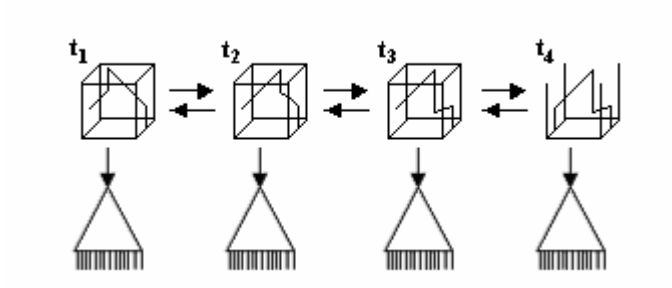

Figura 4: Ejemplo de almacenamiento de una poligonal en diferentes instantes de tiempo

3. CONSULTAS EN EL I+MON-TREE-RM

El soporte de consultas es una de las funciones más importantes de un sistema de información. La respuesta a una consulta depende del tipo de consulta, del sistema de información que administra la consulta y de los índices disponibles para dicho sistema de información.

El índice espacio-temporal I+MON-Tree responde a los principales tipos de consultas, estas son: Instante de tiempo, Trayectoria, Ventana y Rango.

En esta investigación, al plantear una extensión del índice I+MON-Tree vimos que era necesario

realizar un análisis de las modificaciones que deberían efectuarse sobre las consultas que mencionamos anteriormente.

De esta manera, a continuación presentamos cada una de las consultas y señalamos las posibles modificaciones para su adaptación en redes móviles.

Cabe aclarar que, en los diferentes tipos de consultas se mantienen las variables *TMAX3D* y *TMININD* con el objetivo de mejorar la eficiencia en el procesamiento de las consultas evitando en algunos casos el acceso innecesario a alguna de las estructuras.

La variable *TMAX3D* mantiene el tiempo mas reciente registrado en el 3D R-Tree y la variable TMININD mantiene el tiempo más antiguo registrado en el Índice.

CONSULTA POR VENTANA (WINDOW QUERY)

Este tipo de consulta consiste en recuperar todos los objetos que estuvieron o están en un área denominada ventana o sub-espacio, en un intervalo de tiempo dado.

El algoritmo recibe una ventana espacial y un intervalo de tiempo, y busca en el índice, los datos que responden a la consulta.

La ventana de consulta está asociada a un espacio y a un intervalo de tiempo dado, y contiene partes de poligonales, que atraviesan el mismo espacio en esos instantes, incluidos en el intervalo de tiempo. Por tanto, resulta necesario realizar una búsqueda en la estructura I+3 R-Tree para recuperar todos los cubos, los cuáles contienen poligonales, que intersecan con la ventana de consulta.

La manera en que procede la consulta Window Query es la siguiente: identifica las poligonales cuyas posiciones espaciales intersecan el área denominada ventana o sub-espacio en el intervalo de tiempo dado.

Para ello, realiza la búsqueda dentro de la estructura I+3 R-Tree todos los cubos que intersecan con la ventana de consulta. Así, recupera todas las poligonales que están en dicho intervalo de tiempo.

Luego, accede a los R-Tree´s inferiores asociados a cada uno de los cubos que contienen poligonales recuperados anteriormente y recupera aquellos objetos cuyo tiempo de permanencia coincida con alguno de los instantes de tiempo del intervalo. En caso de no haber ocurrido cambios en las posiciones espaciales que conforman las poligonales se realiza la búsqueda dentro de los R-Tree's inferiores correspondientes.

ALGORITMO *WINDOWQUERYI+MON-TREE-RM(R,T_I,T_F,O)*

Entrada: *R* es el rectángulo de consulta, t_i y t_f son el límite inferior y superior respectivamente del intervalo de tiempo.

Salida: Q es el conjunto de objetos que responden la consulta.

1. si t_i>TMAX3D
2. Q´← BuscarEnIndice(p_1,p_2,t_i)
3. Q ← BuscarObjetos(Q´,t_i,t_j)
4. sino
5. region ← armar-región(p1,p2,t_i, t_j)
6. si t_i < TMININD
7. Q´← BuscarEnIndice(p_1,p_2,t_i)´
8. Q´← Q´∪ BuscarEn3D(region)
9. Q ← BuscarObjetos(Q´,t_i,t_j)
10. retornar Q

Donde:

BuscarEnIndice() debido a que todos las poligonales contenidas en el índice se encuentran en su posición actual, por lo tanto el índice solo consta de cubos abiertos y por este motivo solo es necesario hacer la comparación con el piso de los cubos al igual que en consultas de tipo TimeSlice.

BuscarObjetos() recupera aquellos objetos, almacenados en los distintos R-Tree's inferiores correspondientes a los cubos seleccionados, que registraron movimientos en el intervalo de tiempo $[t_i,t_j]$.

armar_region() es un cubo con un piso y un techo dados por los limites inferior (t_i) y superior (t_j) del intervalo de consulta respectivamente.

El algoritmo *BuscarEn3D ()* recibe en este caso, a diferencia que para consultas timeslice, una región de consulta cúbica como parámetro, y devuelve todos los cubos, que contienen poligonales, que se intersecan con éste cubo de consulta.

CONSULTA POR RANGO O INTERVALO (RANGE QUERY)

El procesamiento de este tipo de consulta es muy sencillo. Se siguen los mismos pasos para procesar la consulta Window Query, sólo que se incorpora un paso que permite la eliminación de objetos duplicados, en el conjunto de respuesta.

El algoritmo procede de la siguiente manera: aplica Window Query, así obtiene un conjunto de objetos candidatos que responden a la consulta. Inicializa como vacío el conjunto de respuesta de Range Query. Y para cada uno de los objetos candidatos, si no pertenece al conjunto de respuesta de Range Query, entonces lo incluye al conjunto de respuesta final.

ALGORITMO *RANGEQUERYI+MON-TREE-RM(R,T_I,T_F,O)*

Entrada: *R* es el rectángulo de consulta, t_i y t_f son el límite inferior y superior respectivamente del intervalo de tiempo y *O* es el nº de objetos hallados, inicialmente 0.

Salida: Q es el conjunto de objetos que responden la consulta.

```
1.  WindowQueryI+MON-Tree-RM(R,ti,tf,O)
2. Q´RQ ← ϕ
3. i ← 0
4. mientras i < 0 hacer
5. obj ← BuscarObjeto (Q,i)
6. si (ObjetoNoPertenece(obj,Q´))
7. Q´RQ ← obj
8. O´ ← O´ + 1
9. finsi
10.  i ← i + 1
11.  finmientras
12.  Retornar Q
```

Donde:

BuscarObjeto(Q,i) busca el i-ésimo elemento de Q.

ObjetoNoPertenece(obj,Q´) verifica si obj no pertenece a Q´.

CONSULTA POR INSTANTE DE TIEMPO (TIME SLICE QUERY)

Este tipo de consulta intenta determinar la existencia de objetos que intersequen el espacio consultado en el instante de tiempo dado.

El procesamiento de esta consulta es similar al de Window Query, pero a diferencia de ésta, el argumento recibido no es un intervalo de tiempo sino un instante de tiempo particular.

El espacio de consulta está asociado a un instante de tiempo dado. Éste contiene partes de poligonales, que atraviesan el mismo espacio en ese instante de tiempo.

La manera en que procede la consulta Time Slice Query es la siguiente: identifica las poligonales cuyas posiciones espaciales intersecan el área denominada ventana o sub-espacio en un instante de tiempo dado.

Para ello, realiza la búsqueda dentro de la estructura I+3 R-Tree todos los cubos que intersecan con la ventana de consulta. Así, recupera todas las poligonales que están en dicho instante de tiempo.

Luego, accede a los R-Tree´s inferiores asociados a cada uno de los cubos que contienen poligonales recuperados anteriormente y recupera aquellos objetos cuyo tiempo de permanencia coincida con instante de tiempo correspondiente. En caso de no haber ocurrido cambios en las posiciones espaciales que conforman las poligonales se realiza la búsqueda dentro de los R-Tree's inferiores correspondientes.

ALGORITMO *TIMESLICEQUERYI+MON-TREE-RM(R,T_i,O)*

Entrada: *R* es el rectángulo de consulta, t_i es el instante de tiempo.

Salida: Q es el conjunto de objetos que responden la consulta.

1. si t_i>TMAX3D
2. Q´← BuscarEnIndice(p_1,p_2,t_i)
3. Q ← BuscarObjetos(Q´,t_i,t_i)
4. sino
5. region ← armar-región(p1,p2,t_i, t_i)
6. si t_i < TMININD
7. Q´← BuscarEnIndice(p_1,p_2,t_i)´
8. Q´← Q´∪ BuscarEn3D(region)
9. Q ← BuscarObjetos(Q´,t_i,t_i)
10. retornar Q

Donde:

BuscarEnIndice() debido a que todos las poligonales contenidas en el índice se encuentran en su posición actual, por lo tanto el índice solo consta de cubos abiertos y por este motivo solo es necesario hacer la comparación con el piso de los cubos al igual que en consultas de tipo TimeSlice.

BuscarObjetos() recupera aquellos objetos, almacenados en los distintos R-Tree's inferiores correspondientes a los cubos seleccionados, que registraron movimientos en el tiempo t_i .

armar-region() construye la región de consulta a partir de los puntos que representan los extremos

opuestos de la misma y el tiempo de consulta. Para el caso de una consulta del tipo timeslice la región de consulta es plana, por lo tanto, la tercera coordenada que representa la dimensión temporal tendrá el mismo valor t_i para todo punto en la frontera de la región.

BuscarEn3D() es simplemente el algoritmo de búsqueda del 3D R-Tree, que recibe como parámetro una región plana de consulta, y devuelve todos los cubos, que contienen las poligonales, que se intersecan con esta región de consulta en el tiempo t_i.

CONSULTA POR TRAYECTORIA (TRAJECTORY QUERY)

Este tipo de consulta consiste en recuperar las posiciones espaciales que un objeto fue atravesando en su trayectoria.

Debemos recordar que la recuperación de la trayectoria puede ser de distintos tipos: *parcial*, *completa* y por *intervalo de tiempo*. En la trayectoria parcial se informan las rutas o aristas que atravesó el objeto. En la trayectoria completa se informa todas las posiciones que atravesó el objeto desde el primer al último movimiento informado por el mismo. Y en la trayectoria por intervalo de tiempo recibe como argumento un intervalo temporal, e informa la trayectoria recorrida por ese objeto durante ese intervalo de tiempo.

La manera en que procede la consulta Trajectory Query es la siguiente: recibe el identificador del objeto cuya trayectoria se desea conocer. De acuerdo al tipo de trayectoria, se siguen los siguientes pasos.

En caso de tratarse de una trayectoria parcial, el algoritmo accede a los R-Tree´s inferiores. Allí, la búsqueda se basa en el tiempo y en las posiciones en que el objeto informó su arribo. Con esta información, reconstruye la trayectoria, identificando aquellas aristas que forman parte de las poligonales por las cuales atravesó el objeto.

En caso de tratarse de una trayectoria completa, el algoritmo accede a los R-Tree´s inferiores y recupera la siguiente información: identificador de la poligonal, identificador del objeto, intervalo de tiempo y posición.

Finalmente, en caso de tratarse de una trayectoria por intervalo de tiempo, el algoritmo realiza los mismos pasos que en la trayectoria parcial, con la restricción que la trayectoria del objeto se reconstruye considerando el intervalo de tiempo, es decir, obtiene las posiciones por las cuales transitó el objeto en movimiento en los instantes de tiempo pertenecientes al intervalo.

ALGORITMO *TRAJECTORYQUERY I+MON-TREE-RM(MOID)*

Entrada: *moid* es el identificador del objeto del que se desea recuperar trayectoria.

Salida: Q es el conjunto de movimientos que conforman la trayectoria del objeto.

1. f ← *PrimeroEnTrayectoria(moid)*
2. Q ← ϕ
3. Mientras f ≠ nil hacer //f = nil si no hay mas movimiento en trayectoria
4. Q ← Q ∪ *BuscarMovimientoEnInfoHist(f)*
5. finmientras
6. Retornar Q

Donde:

PrimeroEnTrayectoria(moid) permite recuperar desde Índice de Información Actual, la posición del

primer movimiento que debe ser recuperado, para armar trayectoria.

BuscarMovimientoEnInfoHist(f) recupera el movimiento almacenado en posición f y actualiza f con el próximo movimiento que debe ser recuperado buscando en todas las configuraciones de la misma poligonal a través del puntero *p_cubos*.

4. CONCLUSIONES

En esta investigación presentamos una posible adaptación del índice I+MON-Tree en redes móviles, como así también, hemos planteado las posibles modificaciones que se deberían realizar sobre las consultas que soporta dicho índice.

Es importante notar que esta investigación sólo damos una primera aproximación del diseño de adaptación del índice I+MON-Tree, cuando la red subyacente se trata de una red móvil. Esto implica que, no necesariamente este diseño es el mejor modelo, sino que sólo es una presentación inicial, la cual puede ser modificada y/o reemplazada por otros diseños más adecuados. En sí sirve para ver la complejidad del problema.

Claramente vemos que, no consideramos costos de almacenamiento y tiempo, entre otros. Por lo que creemos haber consolidado y alimentado una línea de estudio, a fines de brindar un puente a futuras investigaciones sobre el tema.

5. REFERENCIAS

[AG04a] Almeida, V. y Güting, R. Indexing the trajectories of moving objects in networks. In *Proc. of the 16th Intl. Conf. on Scientific and Statistical Database Management (SSDBM)*. 2004.

[BRINK00] Brinkhoff, T. Generating Network-Based Moving Objects. In *Proc. of the 12th Intl. Conference on Scientific and Statistical Database Management Berlin, IEEE Computer Society Press.* 2000.

[Cgg06] Carrasco, Fernando D., Gagliardi, Edilma O., García Sosa, Juan C. Una propuesta de un método de acceso espacio-temporal: I+3 R-Tree CACIC 2006. Congreso Argentino de Ciencias de la Computación. 2006.

[Co06] Correa. L, Ortiz. N, I + MON-Tree: índice espacio-temporal para objetos en movimiento; Trabajo final de la Licenciatura en Ciencias de la Computación, *Univ. Nac. de San Luis, Argentina,* 2006. Gagliardi, O., Directora.